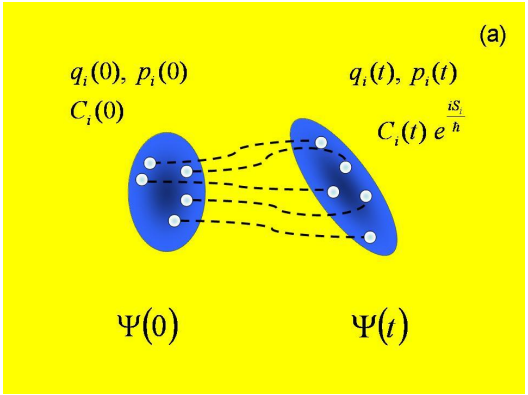
Coupled Coherent States

Code parallelisation and optimisation

Index

CCS code

The quantum dynamics simulations of the CCS code have O(n^2) complexity.



But they still take large amounts of time to run.

Can we do better?

Parallelisation

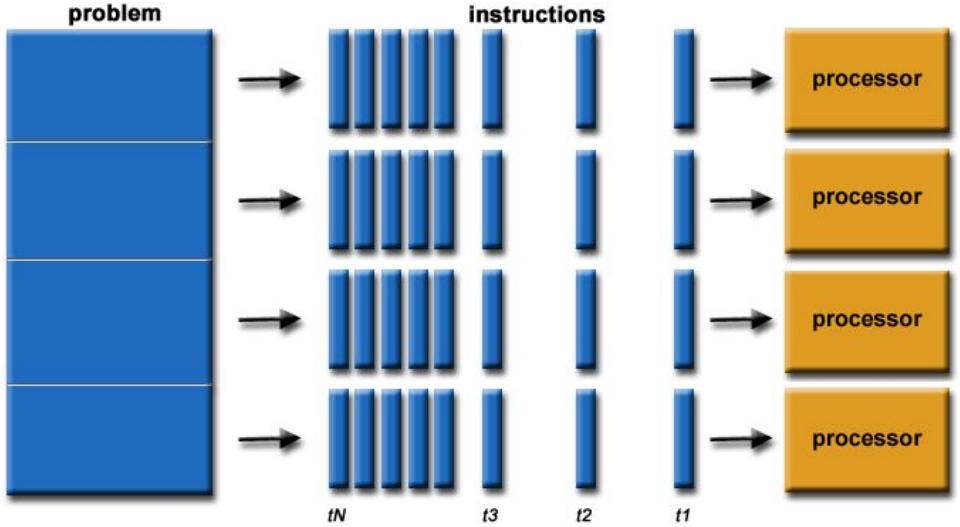
Ideally:

$T_N = T_1 / N$

But in practice:

$T_N = T_1 \ (P + 1/N \times (1-P))$

(Amdahl's law)



Aim workload balance and minimize communications

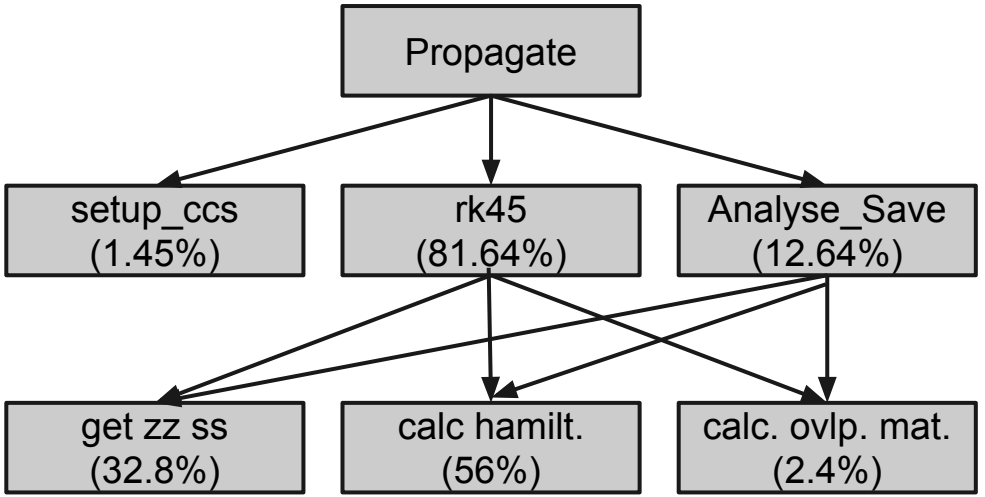
CCS code

- Generate trajectories
  - Monte Carlo algorithm (embarrassingly parallel).
  - Takes less time and once generated they can be used for multiple propagations

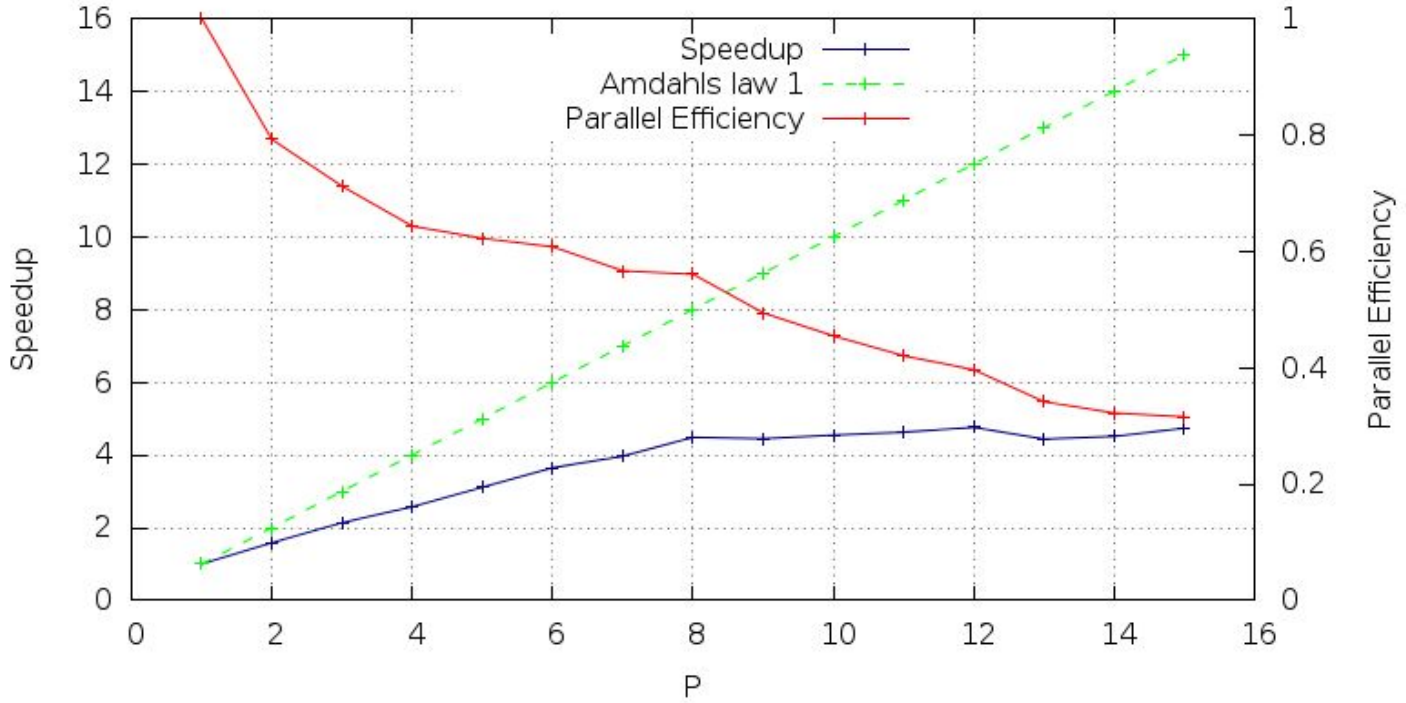
- Propagate trajectories
  - Runge Kutta 45 method
  - Takes most time, challenging to parallelise

Serial Profiling

(32 cs, 20 u.t.)	Time (pgi -O3)
Generate trj. serial	18.429s
Propagate trj. serial	8.85s
Propagate serial (not hermitian)	11.8s (x0.75)
Propagate (save merged)	11.317s
Propagate further optimized	???

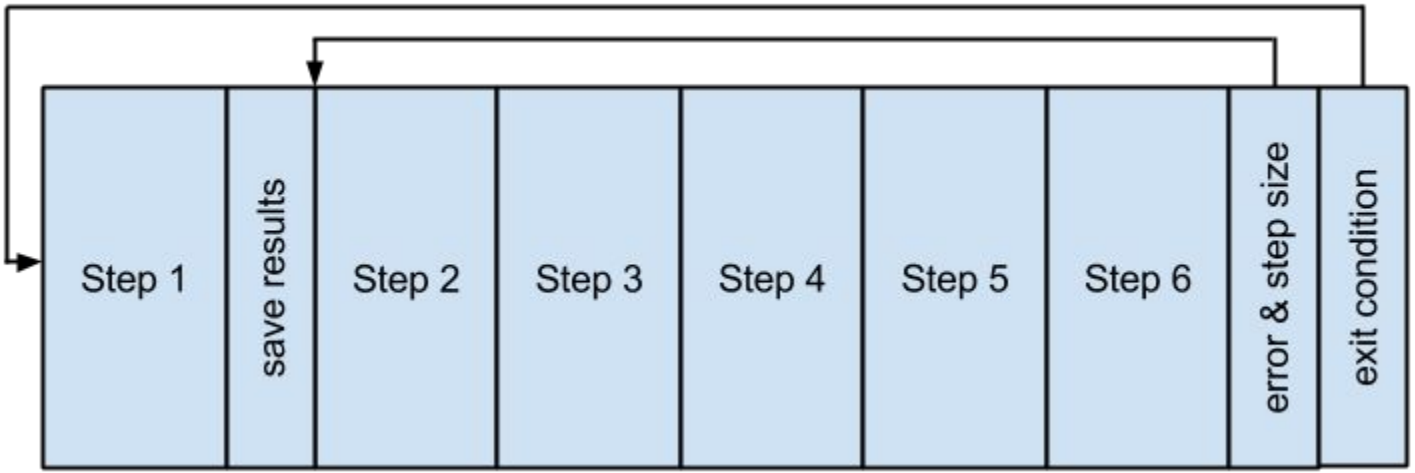


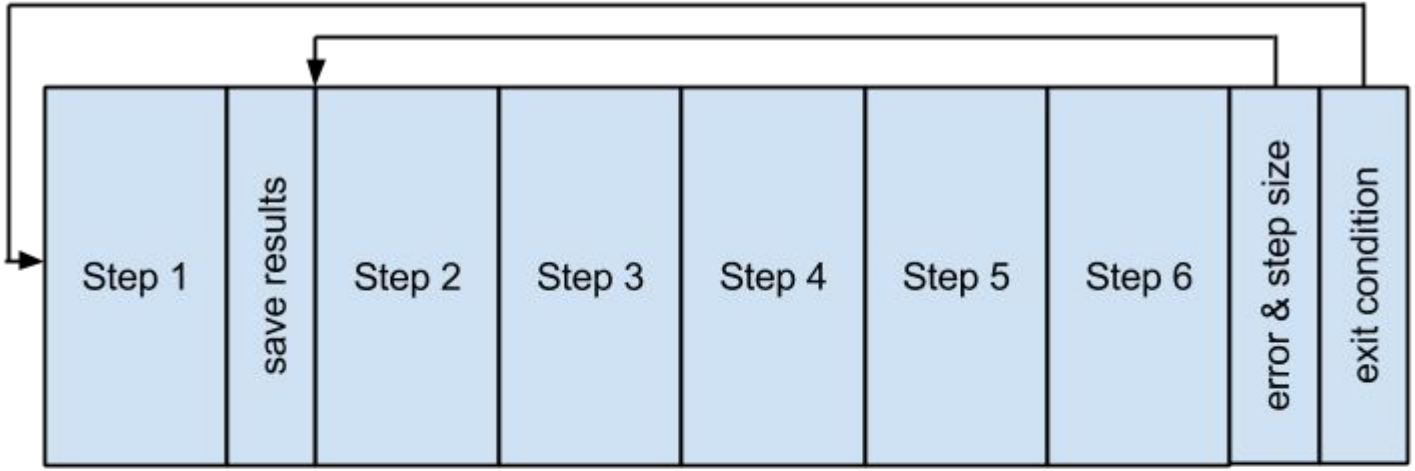
Generate trajectories

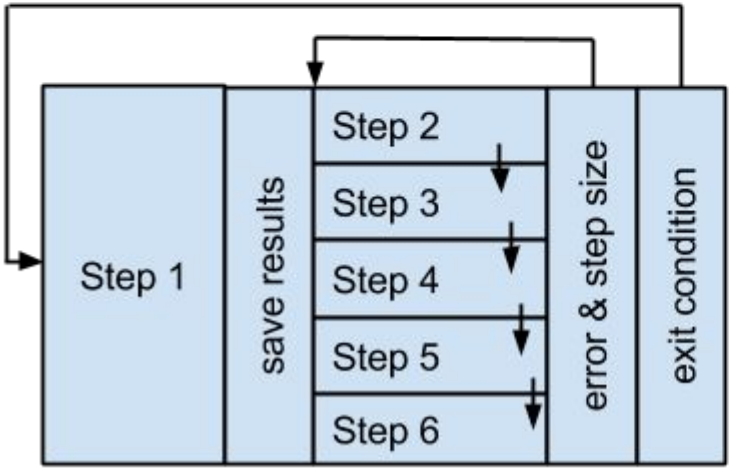


Propagate trajectories

Runge kutta Cash–Karp method





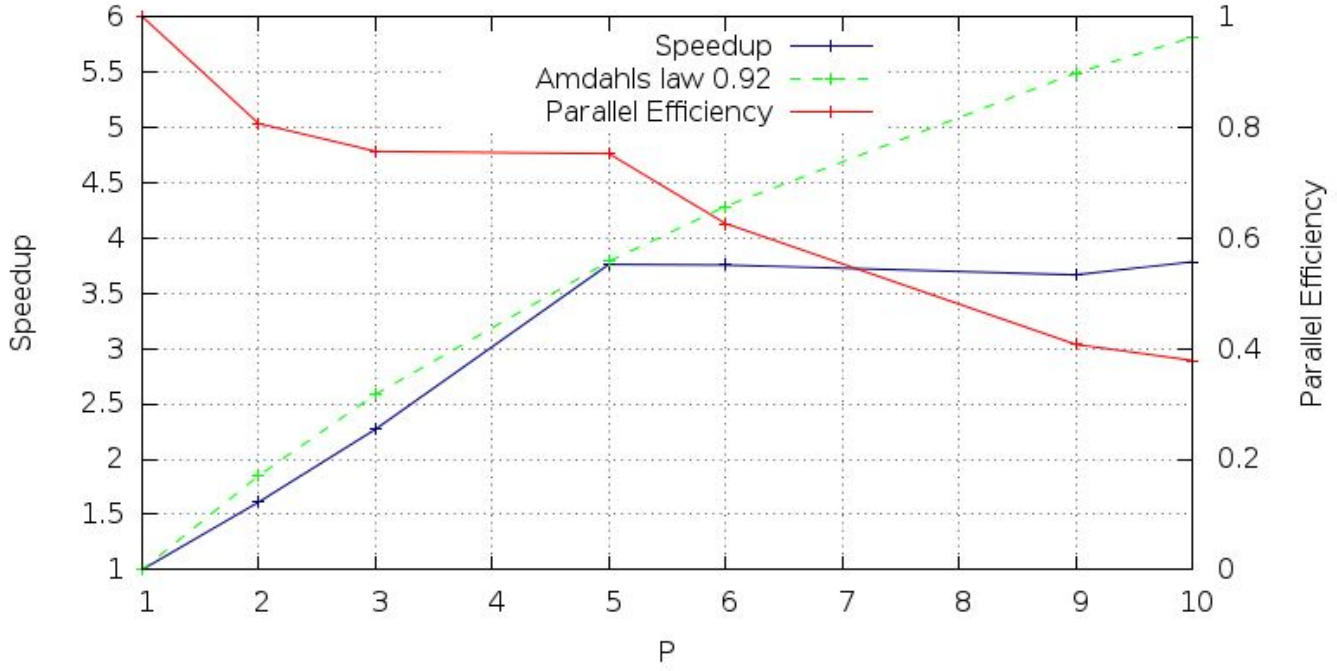


Good load balance, 8 communications of a vector of ncs double complex.

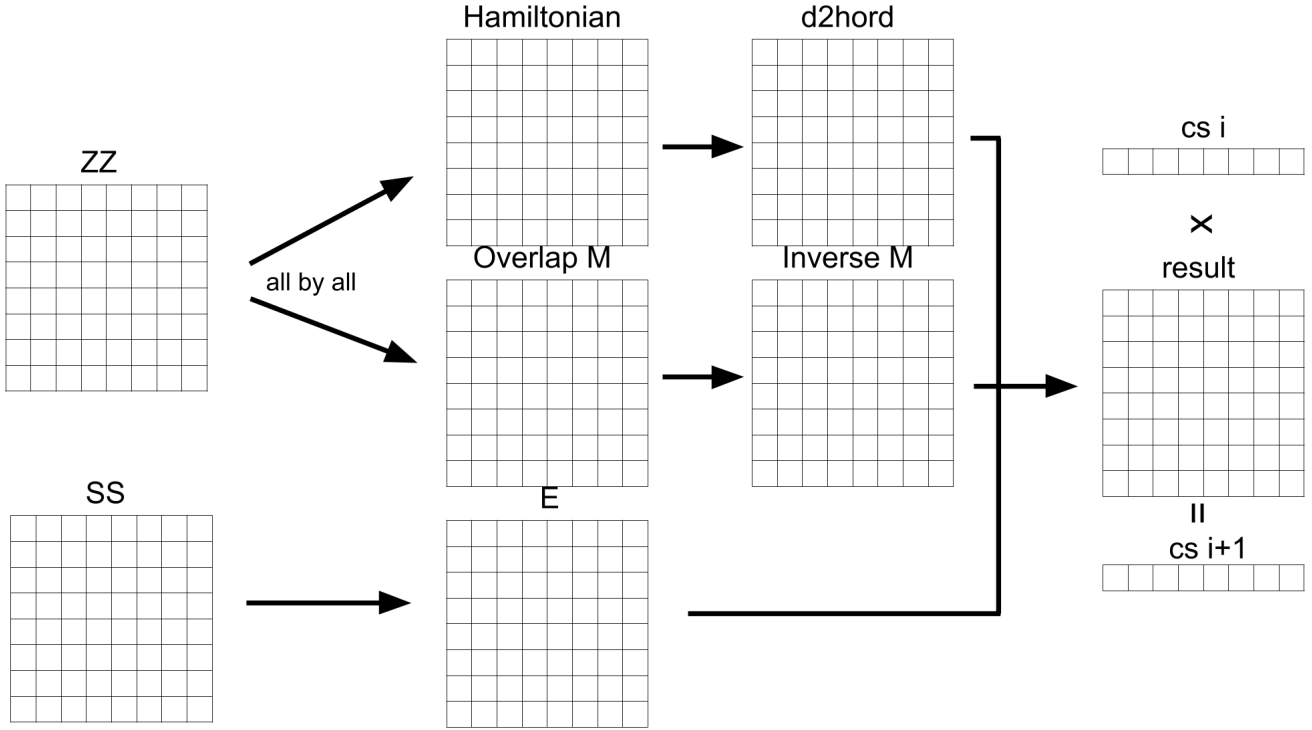
But don’t scales further than 5 processes!!

Expected time:
TN = T1 (1/6 + 1/N x 5/6)

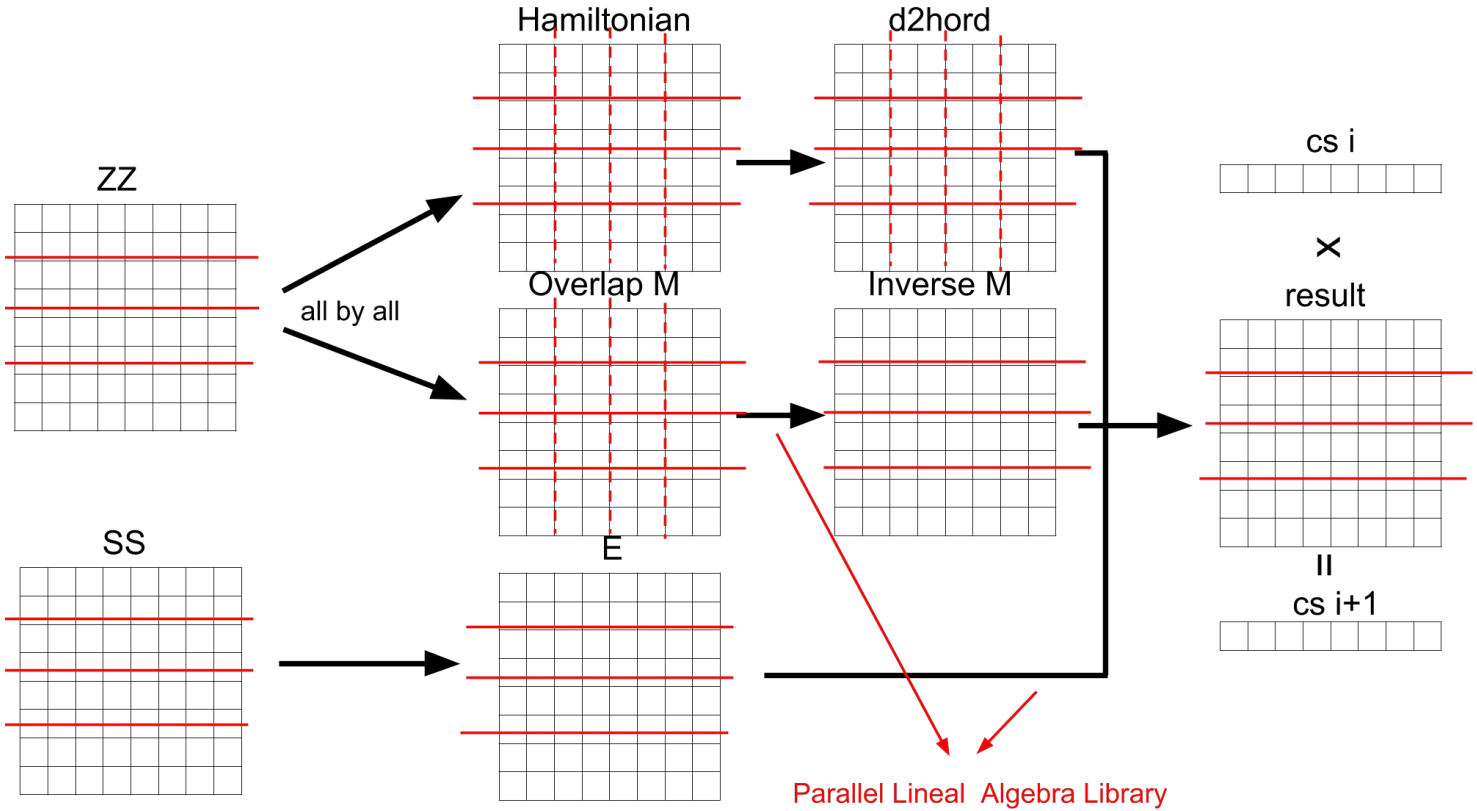
RK45 parallelisation



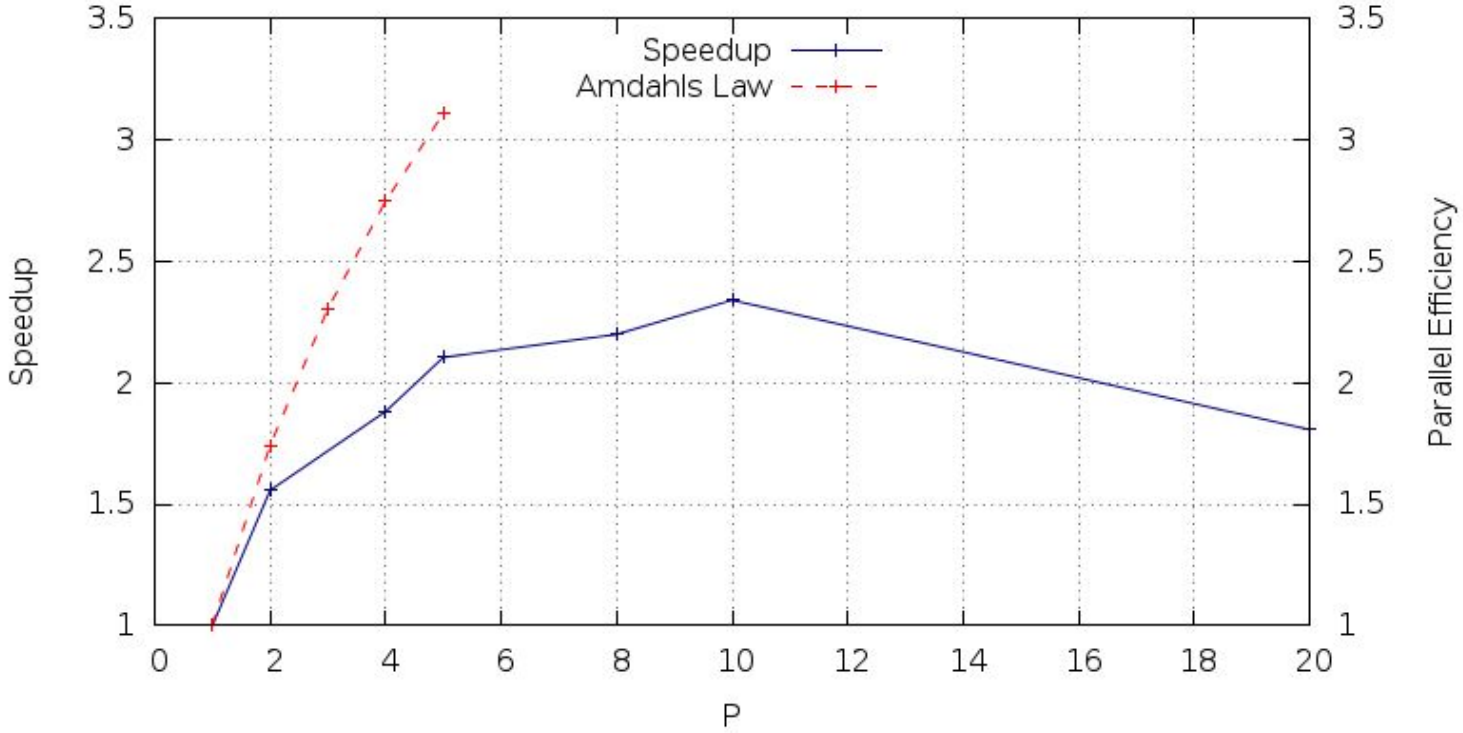
In each step ...


Hamiltonian
d2hord
cs i
ZZ
×
all by all
Overlap M
Inverse M
result
SS
E
II
cs i+1

Good balance if not consider the Hermitian properties.
Lots of communications!
Expected time:



Step parallelisation



Further work

Combine both parallelisations

+

PLASMA LA library (fast for small matrices)

+

Parallel Hermitian calculations

Further scaling

Remove Runge-Kutta 45 synchronizations